Angewandte Mathematik und Informatik

Universität zu Köln

Report No. 01.418

Abuse of Multiple Sequence Alignment in a Paint Shop

by

Th. Epping, W. Hochstättler

Center for Parallel Computing
Universität zu Köln
D–50923 Köln
GERMANY
{epping,wh}@zpr.uni-koeln.de

1991 Mathematics Subject Classification: 90B30, 90C39
Keywords: Dynamic programming, Multiple sequence alignment, Paint shop

Abuse of Multiple Sequence Alignment in a Paint Shop

Th. Epping, W. Hochstättler

July 2, 2001

Abstract

We present a new solution approach for a problem that arises in the automobile industry: the withdrawal of colors from a line storage system such that the resulting number of color changes within the withdrawal sequence is minimized.

We show that in a certain sense this problem is equivalent to the multiple sequence alignment problem known from molecular biology and present some preliminary computational results that indicate the applicability of our approach in practice.

1 Introduction

The paint shop of an automobile plant is usually settled between preceding press and body shops and succeeding assembly lines. It is divided into several booths, in which a car body is cleaned and gets its cathodic immersion painting, its enamel color, and its prime color (see [1]).

Within the enamel booth, a color change occurs whenever two consecutive car bodies of an incoming car body sequence have to be colored in different colors. Thereby, both water pollution and costs arise. Thus, the minimization of the number of color changes within the enamel booth is a desirable objective (see [2]). However, it is unreasonable that the complete production sequencing conforms to this objective, as the resulting car body sequence may be far from optimal for other production phases (e. g. the assembly lines) due to different optimization objectives.

Instead, the usual way to improve on the number of color changes within a car body sequence is the use of an interim storage system, which is placed in front of the enamel booth. Such a system is used to build groups of car bodies that have to be colored in the same color. There exist several kinds of storage systems, which vary with respect to their flexibility and investment costs. Each storage system consists of an input belt, sorting belts and an output belt. The incoming car bodies are spread on the sorting belts according to a set of storage rules. Likewise, an output sequence of car bodies is built according to a set of retrieval rules (see [1]).

In this paper, we focus on the retrieval of car bodies from a line storage system (see Figure 1). Due to their simplicity and low investment costs, these systems are the most common systems in the automobile industry.

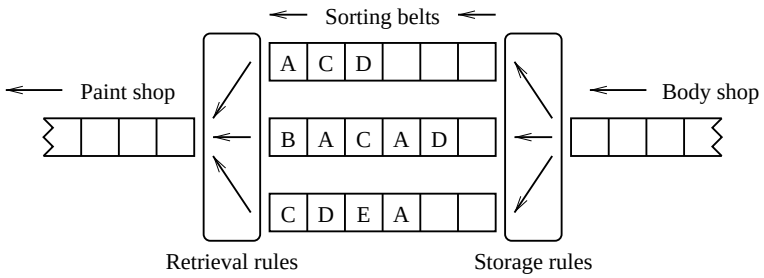

Figure 1: An example of a line storage system. Car body colors on the sorting belts are represented by letters.

In the following, we assume that a specific color is assigned to each car body. Furthermore, we do not distinguish between different car body types and thus identify car bodies with colors.

Definition 1.1 *Given a color set F and a color sequence $R = (R_1, \ldots, R_n)$ with $R_i \in F$ for $i = 1, \ldots, n$, we define*

$$b_j := \begin{cases} 0\,, \text{ if } R_j = R_{j+1} \\ 1\,, \text{ otherwise } (j = 1, \ldots, n-1) \end{cases}$$

We denote the number of color changes within R by $\gamma(R) := \sum_{j=1}^{n-1} b_j$.

Now, we suppose that we are given a snapshot of the line storage system. As we only focus on the retrieval of stored colors, we interpret each sorting belt as a stack. Then, the problem consists in the retrieval of the colors on the stacks such that the number of color changes within the resulting output sequence is minimized.

Problem 1.1 *Color Retrieval Problem (CRP)*

Instance *A finite set $S_1, \ldots, S_s$ of stacks of length $l_1, \ldots, l_s$ that contain colors of a finite color set F.*

Question *Retrieve the colors from $S_1, \ldots, S_s$ in a sequence R such that $\gamma(R)$ is minimized.*

For example, an optimal output sequence with 6 color changes for the configuration depicted in Figure 1 would be

$$R = (B_2 A_1 A_2 C_1 C_2 C_3 A_2 D_1 D_2 D_3 E_3 A_3),$$

where the subscript of a color indicates the sorting belt from which the color has been taken. Note, that we assume that there are no restrictions on the output sequence: we are allowed to permute the incoming car body type sequence arbitrary (within the realms of possibility).

Possible solution approaches to the CRP include the formulation of the CRP as a sequential ordering problem (see [3]) or branch and bound methods. In the following, we show that the CRP is equivalent to the multiple sequence alignment problem in a certain sense. Therefore, we start with a short overview of multiple sequence alignment in the next section.

2 Multiple Sequence Alignment

In molecular biology, multiple sequence alignment (MSA) is used to detect transformations of DNA or protein sequences. We give a short description of MSA, following closely the presentation given in [4].

We start with the two-dimensional case and suppose that we are given two sequences

$$\begin{aligned} a &= (a_1 a_2 \ldots a_n) \\ b &= (b_1 b_2 \ldots b_m) \end{aligned}$$

with $a_i, b_j \in F$, where F is any finite alphabet. We speak of an *alignment* of a and b, if we insert elements $- \notin F$ into a and b, such that we get sequences a' and b' that both have the same length L with $\max\{n, m\} \leq L \leq n + m$. We interpret a' and b' as the rows of a $(2 \times L)$-alignment matrix

$$A = \begin{pmatrix} a'_1 & a'_2 & \ldots & a'_L \\ b'_1 & b'_2 & \ldots & b'_L \end{pmatrix}$$

with $a'_i, b'_j \in F \cup \{-\}$.

Next, we introduce a distance function

$$d : (F \cup \{-\})^2 \to \mathbb{N}_{\geq 0} \cup \infty.$$

and define

$$D(a,b) := \min_{A \in \mathcal{A}(a,b)} \sum_{i=1}^{L} d(a'_i, b'_i)$$

to be the minimal value of an alignment of a and b, where $\mathcal{A}(a,b)$ denotes the set of all alignments of a and b. The value $D(a,b)$ can be computed with a space and time complexity of $\mathcal{O}(nm)$ by the dynamic program shown in Figure 2. The dynamic program can be easily extended to find the optimal alignment itself.

1. `Define`

$$D_{i,j} := D(a_1 a_2 \ldots a_i, b_1 b_2 \ldots b_j)$$

`and set`

$$\begin{aligned} D_{0,0} &= 0 \\ D_{0,j} &= \sum_{k=1}^{j} d(-, b_k) \\ D_{i,0} &= \sum_{k=1}^{i} d(a_k, -) \end{aligned}$$

2. `For` $i = 1$ `to` n
   `For` $j = 1$ `to` m

$$D_{i,j} = \min\{D_{i-1,j} + d(a_i, -), D_{i-1,j-1} + d(a_i, b_j), D_{i,j-1} + d(-, b_j)\}$$

Figure 2: The dynamic program for an alignment of two sequences.

The notation and the results of the two-dimensional case can be extended to deal with the more general case, in which we are given r sequences

$$\begin{aligned} a_1 &= (a_{1,1} a_{1,2} \ldots a_{1,n_1}) \\ a_2 &= (a_{2,1} a_{2,2} \ldots a_{2,n_2}) \\ &\vdots \\ a_r &= (a_{r,1} a_{r,2} \ldots a_{r,n_r}) \end{aligned}$$

with $a_{i,j} \in F$.

We know that the MSA problem is $\mathcal{NP}$-complete if we use the sum of pairs score (which we define in Section 3) as a distance function (see [5]), and that any MSA instance can be solved by a generalization of the dynamic program shown in Figure 2, requiring a space complexity of $\mathcal{O}(\Pi_{i=1}^{r} n_i)$ and a time complexity of $\mathcal{O}(2^r \Pi_{i=1}^{r} n_i)$ (see [4]). There exist implementations of several algorithms to solve an MSA instance to optimality. In Section 3.1, we discuss some preliminary computational results.

In the following, we assume that an alignment of r sequences is given in the form of an $(r \times L)$-alignment matrix

$$A = \begin{pmatrix} a'_{11} & a'_{12} & \dots & a'_{1L} \\ a'_{21} & a'_{22} & \dots & a'_{2L} \\ \vdots & & & \\ a'_{r1} & a'_{r2} & \dots & a'_{rL} \end{pmatrix}$$

with $a'_{ij} \in F \cup \{-\}$.

3 Abuse of Multiple Sequence Alignment

In this section we show that the CRP and the MSA problem are equivalent in a certain sense. Recall that we are given a finite set $S_1, \dots, S_s$ of stacks. Each stack contains colors of a finite color set F. We interpret each stack as an input sequence for the MSA problem and want the resulting alignment matrix A to give a unique and optimal withdrawal sequence for the colors contained in the stacks.

Therefore, we first demand that no stack contains a subsequence of two or more consecutive identical colors and merge any subsequence of that form into a single color element (see Figure 3). Correspondingly, we assume in the following that each color on a stack is framed by different colors.

Proposition 3.1 *The merging of consecutive identical colors does not increase the minimal number of color changes for an instance of the CRP.* □

Next, the main idea is to calculate an optimal MSA matrix A and to derive an optimal withdrawal sequence R for the colors on the stacks by flattening A column by column into the sequence

$$R = (a'_{11}a'_{21} \dots a'_{r1}a'_{12}a'_{22} \dots a'_{r2} \dots a'_{1L}a'_{2L} \dots a'_{rL}),$$

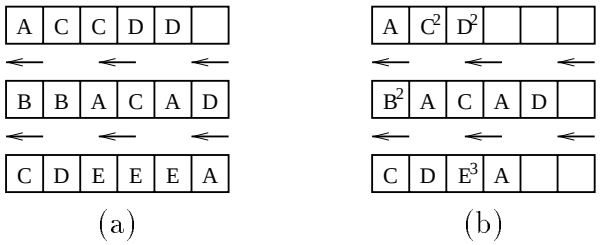

Figure 3: An example CRP instance (a) before and (b) after merging consecutive identical colors. The superscript indicates the multiplicity of a color.

assuming that each stack is open to the left. If we pass through R from the left to the right, we interpret an element $a'_{ij} = f$ as to withdraw color f from stack i and an element $a'_{ij} = -$ as to withdraw nothing from stack i. For the calculation of A, we need to introduce a distance function for an MSA before we can apply the dynamic program mentioned in Section 2. We use the sum of pairs score (see [4]) defined by

$$c(A) := \sum_{i<j} \sum_{l=1}^{L} d(a'_{i,l}, a'_{j,l}),$$

and define the distance function for two sequence elements by

$$d(x,y) := \begin{cases} 0 & , \text{if } x = y \\ 1 & , \text{if } x = - \text{ or } y = - \\ \infty & , \text{if } x \neq y \end{cases}$$

for $x, y \in F \cup \{-\}$.

Note, that these settings prevent that any column of A contains more than one color and force A to contain as few columns as possible. Hence, we get our first lemma.

Lemma 3.1 *Suppose that the solution of a CRP instance results in an optimal MSA matrix A and a corresponding withdrawal sequence R. Then, R contains $\gamma(R) = L - 1$ color changes.*

Proof The definition of the distance function d, the usage of the sum of pairs score and Proposition 3.1 make sure that we get a color change in R if and only if we switch from one column of A to the next. □

On the other hand, we can construct an MSA instance from an arbitrary withdrawal sequence.

Lemma 3.2 *Suppose that we are given a withdrawal sequence R that contains $L-1$ color changes. Then, we can construct a corresponding MSA matrix A with L columns and r rows, where r is the length of a longest subsequence of consecutive identical colors in R.*

Proof For the construction of A, we pass through R from the left to the right, write each consecutive subsequence of identical colors row by row in a different column, and assign the element $-$ to all undefined positions of A. □

Finally, we prove that our solution approach solves an instance of the CRP to optimality.

Lemma 3.3 *Suppose that we solve an instance I of the CRP as described above, yielding an MSA matrix A. Then, the flattening of A gives an optimal withdrawal sequence R for I.*

Proof Suppose that R is not optimal and that there exists a withdrawal sequence R' such that $\gamma(R') < \gamma(R)$. Using Lemma 3.2, we can construct from R' a corresponding MSA matrix A' that contains less columns than A. Note, that we can construct A' such that A' and A contain the same number of rows: due to the merging of consecutive identical colors on each stack of I, the length of a longest subsequence of consecutive identical colors in R' can not exceed the number of stacks. Then, by the definition of our distance function d and the usage of the sum of pairs score, the fact that A' contains less columns than A contradicts the optimality of A. □

3.1 Computational results

In practice, the cycle time for the car body storage in a line storage system ranges from 30 to 60 seconds. Thus, the CRP has to be solved in real-time.

We assume that a line storage system usually consists of $s \leq 6$ sorting belts, each with a capacity of $l \leq 20$ car bodies, while the underlying color set has a cardinality of $|F| \leq 20$. We used the implementation (see [6]) of an improved A^* algorithm (see [7]) for the dynamic programming solution of MSA instances to test the applicability of our solution approach. Table 1 shows some preliminary computational results for randomly created instances of the CRP with completely filled sorting belts.

All examples were run on a SUN E450 with 4 SunUltraII 400 MHz processors and 1152 MB memory. The given running times are averaged over 10

	$l = 5$	$l = 10$	$l = 15$	$l = 20$
$\|F\| = 5$	0.07	0.20	0.57	1.67
	(0.05/0.08)	(0.12/0.42)	(0.34/0.82)	(0.41/5.08)
$\|F\| = 10$	0.07	0.22	1.83	5.64
	(0.04/0.09)	(0.11/0.51)	(0.25/11.79)	(0.75/22.82)
$\|F\| = 15$	0.07	0.18	2.76	5.45
	(0.04/0.10)	(0.13/0.27)	(0.59/6.72)	(1.18/12.60)
$\|F\| = 20$	0.07	0.51	1.76	15.65
	(0.05/0.10)	(0.11/2.60)	(0.54/4.17)	(3.27/40.83)

Table 1: Running times in seconds for the solution of randomly created CRP instances.

instances. Additionally, the best and worst running times are given. Note, that the running time is influenced by the cardinality of the color set and may vary considerably between different instances with identical parameter settings.

In practice, sorting belts are never filled completely. Furthermore, the implementation of the improved A^* algorithm uses the LEDA library (see [8]) which imposes a significant time and space overhead. Thus, Table 1 indicates the applicability of our approach in practice.

References

[1] Sven Spieckermann and Stefan Voß: *Paint Shop Simulation in the Automotive Industry.* ASIM Mitteilungen 54 (1996), pp. 367-380.

[2] Thomas Epping, Winfried Hochstättler and Peter Oertel: *Some Results on a Paint Shop Problem for Words.* Technical report zaik-2001-413, Center of Applied Computer Science, University of Cologne, Germany, 2001. Extended abstract, submitted to: Electronic Notes in Discrete Mathematics.

[3] Norbert Ascheuer: *Hamiltonian Path Problems in the Online-Optimization of flexible manufacturing systems.* Ph. D. Thesis, University of Technology Berlin, Germany, 1995.

[4] Michael S. Waterman: *Introduction to Computational Biology.* Chapman & Hall, 1995.

[5] Lusheng Wang and Tao Jiang: *On the Complexity of Multiple Sequence Alignment.* Journal of Computational Biology, Volume 1, Number 4, 1994, pp. 227-348.

[6] ftp://ftp.mpi-sb.mpg.de/pub/outgoing/reinert/GSA.tgz

[7] Martin Lermen and Knut Reinert: *The Practical Use of the $\mathcal{A}^*$ Algorithm for Exact Multiple Sequence Alignment.* Technical Report 97-1-028, Max-Planck-Institut für Informatik, Saarbrücken, Germany, 1997.

[8] Kurt Mehlhorn and Stefan Näher: *LEDA, a platform for combinatorial and geometric computing.* Communications of the ACM, 38(1):96-102, 1995.